



# Integrated Autonomous Network Management (IANM) Multi-Topology Route Manager and Analyzer

## Final Report

**Contract:** Office of Naval Research (ONR) N00014-05-C-0012  
(ONR Program Officer: Adrian Eley)

**CDRL:** 004

**Organization:** Boeing Phantom Works, Networked Systems Technology  
(Boeing Program Manager: Jae H. Kim)

**Prepared by:**  
Thomas R. Henderson

**Contributors:**  
Kyle Bae  
Jin Fang  
David M. Kushi

**Date:** February 2008

### DISTRIBUTION STATEMENT A.

Approved for public release; distribution is unlimited.

Copyright © 2008, The Boeing Company.

The Boeing Company  
P.O.Box 3707, MC 7L-49  
Seattle, WA 98124

Boeing and Boeing Phantom Works are trademarks of The Boeing Company in the United States, other countries, or both. Other company, product, or service names may be trademarks or service marks of others.

Report Documentation Page			Form Approved OMB No. 0704-0188		
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE <b>FEB 2008</b>		2. REPORT TYPE		3. DATES COVERED <b>00-00-2008 to 00-00-2008</b>	
4. TITLE AND SUBTITLE <b>Integrated Autonomous Network Management (IANM) Multi-Topology Route Manager and Analyzer</b>			5a. CONTRACT NUMBER		
			5b. GRANT NUMBER		
			5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S)			5d. PROJECT NUMBER		
			5e. TASK NUMBER		
			5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) <b>The Boeing Company,P.O. Box 3707. MC 7L-49,Seattle,WA,98124</b>			8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSOR/MONITOR'S ACRONYM(S)		
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)		
12. DISTRIBUTION/AVAILABILITY STATEMENT <b>Approved for public release; distribution unlimited</b>					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT <b>Same as Report (SAR)</b>	18. NUMBER OF PAGES <b>37</b>	19a. NAME OF RESPONSIBLE PERSON
a. REPORT <b>unclassified</b>	b. ABSTRACT <b>unclassified</b>	c. THIS PAGE <b>unclassified</b>			

This page intentionally left blank.

## Table of Contents

Executive Summary.....	1
1. Introduction .....	1
1.1 Summary of Accomplishments.....	2
1.2. Terminology .....	3
1.2 Methods, Assumptions, and Procedures .....	3
2. Concept of Operations .....	4
2.1 MTR Overview.....	4
2.1.1 Components of MTR .....	5
2.2 MTR and IANM .....	5
2.3 Policy-Based Routing as an MTR Approximation .....	7
3. Task 1 - System Engineering .....	9
3.1 Software Architecture.....	9
3.1.1 MTR Data Collection Engine .....	10
3.1.2 MTR Planning Tool .....	10
3.2 Software Development .....	11
4. Task 2 - MTR Data Collection Engine .....	13
4.1 MTR Agent.....	13
4.2 NetFlow Collector .....	13
5. Task 3 - MTR Basic Analysis.....	15
5.1 System Software Flow .....	15
5.2 Development of a Routing Plan.....	18
5.2.1 Inputs .....	18
5.2.2 Outputs.....	19
5.2.3 Calculation.....	19
5.2.4 Generating MTR Link Metrics .....	19
5.2.5 Implementation.....	20
6. Task 4 - MTR Analysis with Policy Constraints .....	22
6.1 Goals.....	23
6.2 Software Overview .....	25
6.3 Trident Warrior 2008 LOE Testing .....	26
7. Results, Discussion, and Conclusions.....	28
References .....	29
Appendix A: Memorandum of Software Transfer	

This page intentionally left blank.

## Executive Summary

This report documents the Multi-Topology Routing (MTR) Route Manager and Analyzer (RMA) program for The Boeing Company under Office of Naval Research (ONR) Contract N00014-05-C-0012. The MTR RMA program ran from July 2005 through February 2008, and was part of a broader Integrated Autonomous Network Management (IANM) program funded by ONR and coordinated by Navy SPAWAR Systems Center, San Diego.

The goal of Boeing's IANM MTR Route Manager and Analyzer (RMA) project was to explore the feasibility of exploiting Multi-Topology Routing (MTR) or similar technologies to perform load balancing and traffic engineering in Navy ADNS scenarios. In a previous ONR research effort, Boeing and Cisco Systems had studied the applicability of MTR in the context of Navy network scenarios, and Boeing had produced a Linux-based prototype of MTR-enhanced routing software for the Open Shortest Path First version 2 (OSPFv2) protocol. While the previous effort demonstrated the utility of MTR (which was later highlighted by a successful coalition Trident Warrior 2006 Sea Trial), a key missing component was an analytical tool that could assist the operator in generating robust and efficient routing configurations for larger networking topologies. Hence, Boeing proposed and ONR selected for funding the Multi-Topology Routing Route Manager and Analyzer (RMA) program. Boeing was instructed by ONR to coordinate its efforts with the IANM program, and Boeing therefore initially focused on building the MTR RMA capability as an IANM component.

Functionally, the MTR RMA is a network management process that gathers and analyzes network topology and traffic statistics to generate a policy-based routing plan based on an operator's goals and policies. The anticipated payoff is an improved packet routing capability for using all available RF bandwidth efficiently, in a manner that is fault tolerant, that is easy to manage and assess its effectiveness, and that respects policy constraints in the network configuration.

Boeing's program made the following contributions:

1) developed research software, known as the MTR Planner, that can perform basic load balancing of an OSPF-based network, given a set of traffic inputs on the system. We developed techniques to leverage Cisco NetFlow information to dynamically determine the traffic matrix imposed on a network. We also developed interfaces and techniques to allow the MTR Planner to passively observe routing updates in a network, and to directly interact with Cisco routers to upload configuration changes and download network management information. Because Cisco routers unexpectedly did not support OSPF MTR routing during the timeframe of this contract, we developed a workaround that allowed our system to produce route maps that functionally replaced the MTR routing capability. Finally, we performed the system engineering necessary to integrate all of the software into running demonstration systems.

We developed two variants of this software: a *centralized planner* integrated with other IANM software, and a *distributed planner* decoupled from other IANM software.

- The *centralized planner* provided an interface to allow mission plans to be distilled into low-level routing configurations deployed globally in the network (to IANM agents operating on individual ships). We developed a capability that not only supported this off-line planning scenario but also could be used in a real-time network monitoring and maintenance capacity. This capability was integrated with the other IANM software components and demonstrated in conjunction with the broader IANM system.
- The *distributed planner* manages a set of outbound links from the router to which it is connected. This version of the software has no other IANM dependencies. It is again designed to work with the Cisco routers being deployed in coalition and ADNS networks. It operates autonomously from ship to ship, which limits the scope of its effectiveness (a given ship can only control its outbound links, and global coordination is not possible) but offers deployment advantages.

2) supported the broader IANM program by participating in two integrated demonstrations at SPAWAR Systems Center, San Diego (March 2006 and August 2006) and in an MTR-specific demonstration in February 2007. In addition, we supported a Trident Warrior 2008 Limited Objective Experiment (LOE) in January 2008 with our distributed planner.

3) supported the MTR experiment as part of the Allied Multi Bearer Routing Networking (AMBTN) initiative during Trident Warrior 2006. Coalition ships were equipped with a Linux-based router running Boeing's MTR software. Results from testing showed that all packets were marked and routed correctly, while the network

administrative traffic load was reduced.

4) participated in the standardization effort for MTR by contributing to the following Internet-Drafts at the Internet Engineering Task Force (IETF):

- *Multi-Topology (MT) Routing in OSPF* (now RFC 4915)
- *OSPF Version 2 MIB for Multi-Topology (MT) Routing* (Internet Draft: draft-ietf-ospf-mt-mib-01.txt)

5) published a technical paper in IEEE Milcom 2007 conference: *Traffic Engineering with OSPF Multi-Topology Routing*.

At the request of ONR/SPAWAR, Boeing made two key technical adaptations in the course of this work; both were judged to be covered under the technical scope of the statement of work. The first was an adaptation of the originally proposed software to become compatible with the overall IANM system. Boeing's proposed system design called for a standalone traffic management tool deployed at a Network Operations Center (NOC), useful for dynamic monitoring and reconfiguration of an operational network. IANM developed a service-oriented architecture (SOA) with components operating autonomously on-board ships, and had an overall focus on mission planning and not real-time network adjustment. Boeing adapted its design in several ways, including providing an interface to a higher-layer IANM planning tool, using and extending the IANM database for storing its information, running portions of its software as an IANM system component, and integrating with the SFA-developed XCAFE architecture and graphical user interface (e.g., as an XCAFE plug-in). In 2007, we were instructed to decouple our software from IANM and to focus on support of coalition networking scenarios (outside of IANM deployment plans). This again caused us to refactor our software appropriately; in particular, to become more distributed in operation and to focus again on dynamic, real-time monitoring and response.

The second major change was to compensate for the unexpected lack of support for MTR in Cisco router implementations. Due to the program's desire to use Cisco routers as being deployed in the fleet, Boeing redesigned the software to output routing configuration in the form of policy-based route maps instead of the planned MTR link metrics. We briefly describe in this report the consequences of this change.

This document provides a summary of the key milestones and accomplishments of the MTR RMA project, with pointers to more detailed documentation. In 2008, Boeing participated in a Trident Warrior 2008 Limited Objective Experiment (LOE) and transferred the software to SSC San Diego, where it may be used in future coalition networking experiments. This transfer was documented in a memorandum dated 10 January 2008 and signed by the IANM, SSC SD, and ONR representatives.

## 1. Introduction

This report documents the Multi-Topology Routing (MTR) Route Manager and Analyzer (RMA) program for The Boeing Company under Office of Naval Research (ONR) Contract N00014-05-C-0012. The MTR RMA program ran from July 2005 through February 2008, and was part of a broader Integrated Autonomous Network Management (IANM) program funded by ONR and coordinated by Navy SPAWAR Systems Center, San Diego.

The goal of Boeing's IANM MTR Route Manager and Analyzer (RMA) project was to explore the feasibility of exploiting Multi-Topology Routing (MTR) or similar technologies to perform load balancing and traffic engineering in Navy ADNS scenarios. From the Statement of Work [Boe05]:

“The contractor will conduct research on the development of the Integrated Autonomous Network Management Multi-Topology Route Manager and Analyzer. The contractor shall develop a multi-topology routing (MTR) management capability that plugs into the overall Integrated Autonomous Network Management (IANM) framework. The contractor shall coordinate its efforts under the overall direction of the IANM lead at the Space and Naval Warfare Systems Command (SPAWAR).”

In a previous ONR research effort, Boeing and Cisco Systems had studied the applicability of MTR [RFC4915] in the context of Navy network scenarios, and Boeing had produced a Linux-based prototype of MTR-enhanced routing software for the Open Shortest Path First version 2 (OSPFv2) protocol [RFC2328]. While the previous effort demonstrated the utility of MTR (which was later highlighted by a successful coalition Trident Warrior 2006 Sea Trial), a key missing component was an analytical tool that could assist the operator in generating robust and efficient routing configurations for larger networking topologies. Hence, Boeing proposed and ONR selected for funding the Multi-Topology Routing Route Manager and Analyzer (RMA) program. Boeing was instructed by ONR to coordinate its efforts with the IANM program, and Boeing therefore initially focused on building the MTR RMA capability as an IANM component.

Functionally, the MTR RMA is a network management process that gathers and analyzes network topology and traffic statistics to generate a policy-based routing plan based on an operator's goals and policies. The anticipated payoff is an improved packet routing capability for using all available RF bandwidth efficiently, in a manner that is fault tolerant, that is easy to manage and assess its effectiveness, and that respects policy constraints in the network configuration.

At the request of ONR/SPAWAR, Boeing made two key technical adaptations in the course of this work; both were judged to be covered under the technical scope of the statement of work. The first was an adaptation of the originally proposed software to become compatible with the overall IANM system. Boeing's proposed system design called for a standalone traffic management tool deployed at a Network Operations Center (NOC), useful for dynamic monitoring and reconfiguration of an operational network. IANM developed a service-oriented architecture (SOA) with components operating autonomously on-board ships, and had an overall focus on mission planning and not real-time network adjustment. Boeing adapted its design in several ways, including providing an interface to a higher-layer IANM planning tool, using and extending the IANM database for storing its information, running portions of its software as an IANM system component, and integrating with the SFA-developed XCAFE architecture and graphical user interface (e.g., as an XCAFE plug-in). In 2007, we were instructed to decouple our software from IANM and to focus on support of coalition networking scenarios (outside of IANM deployment plans). This again caused us to refactor our software appropriately; in particular, to become more distributed in operation and to focus again on dynamic, real-time monitoring and response.

The second major change was to compensate for the unexpected lack of support for MTR in Cisco router implementations. Due to the program's desire to use Cisco routers as being deployed in the fleet, Boeing redesigned the software to output routing configuration in the form of policy-based route maps instead of the planned MTR link metrics. We briefly describe in this report the consequences of this change.

In 2008, Boeing participated in a Trident Warrior 2008 Limited Objective Experiment (LOE) and transferred the software to SSC San Diego, where it may be used in future coalition networking experiments. This transfer was documented in a memorandum dated 10 January 2008 and signed by the IANM, SSC SD, and ONR representatives (see Appendix A).



## 1.1 Summary of Accomplishments

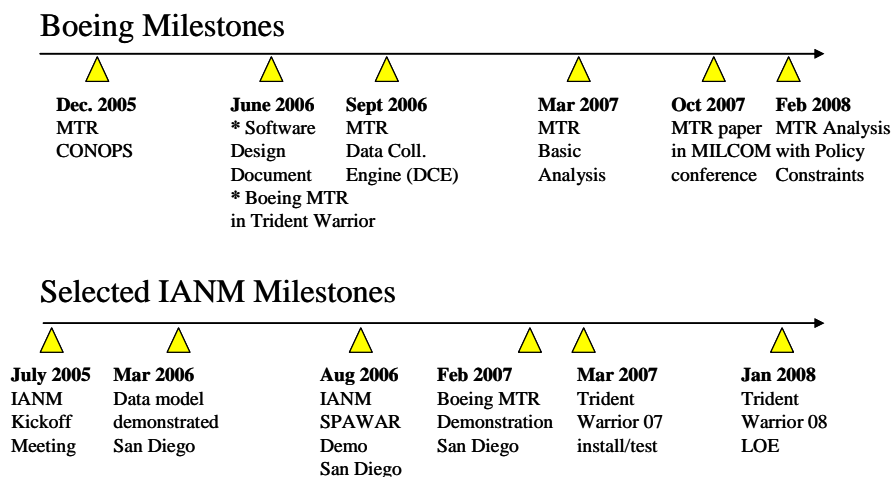


Figure 1-1. Timeline of key events in the Boeing IANM MTR-RMA project.

Figure 1-1 illustrates a timeline of key events in the Boeing IANM MTR-RMA project. The above figure, and below table, summarize the contributions of this project. Cited references are listed at the end of this report.

Milestone	Date	Description	Contract Deliverable	Reference
IANM Kickoff Meeting	July 2005	Program kickoff meeting	Slides	N/A
MTR CONOPS document	Dec 2005	Technical report that described how MTR extensions apply to the Concept of Operations (CONOPS) for the Automated Digital Network System (ADNS).	No	[HKB+05]
Data model demonstrated	Mar 2006	Spiral 1.0b demonstration, SSC-San Diego	No	N/A
Software Design Document	June 2006	Technical report describing the system design and software architecture	CDRL A003	[Boe06]
Boeing MTR in Trident Warrior 2006	June 2006	MTR software supported the coalition networking tests in Trident Warrior 2006	No	[Sib06]
IANM SPAWAR Demo	Aug 2006	IANM program demonstration, SSC-San Diego	No	N/A
MTR Data Collection Engine	Sept 2006	Software module used to collect and publish statistical information from routers	CLIN 0002	[BHK06]
Boeing MTR Demonstration	Feb 2007	MTR was separately demonstrated from IANM (Trident Warrior 07) since IANM did not have multiple ships or access to the routers in TW07	No	N/A
MTR Basic Analysis	Mar 2007	Software module that publishes routing configuration based on the communications plan and traffic inputs	CLIN 0003	[BHK07]
MTR Milcom paper	Oct 2007	Technical publication on the applicability of route management to MTR-enabled networks	No	[Bae07]
Trident Warrior LOE testing	Jan 2008	Testing of the distributed planner software in a coalition networking scenario	No	N/A
MTR Analysis with Policy Constraints	Feb 2008	Software module implementing the distributed planner that locally manipulates route maps	CLIN 0004	[Hen08]

Table 1-1. Description of key milestones depicted in Figure 1-1.

**Note:** This report constitutes CDRL A004 of the contract. CDRL A001 (Monthly Reports) and CDRL A002 (Program Reviews) are not specially called out above.

## **1.2. Terminology**

### **Multi-Topology Routing (MTR):**

An OSPF extension that allows for multiple logical topologies to be defined on an underlying single physical topology, for the purpose of computing different shortest path trees in the network for different traffic types. The Multi-Topologies are represented by different metric values in the Type-of-Service (TOS) metric fields within the routing link state advertisements (LSAs).

### **Multi-Topology Routing using Route Maps (MTR-RM):**

An approximation to MTR routing using policy-based route maps. Policy-based route maps are a form of static source routing that bypasses (overlays) the routes built by a dynamic routing protocol such as OSPF. They provide finer routing granularity than does destination-based routing based on OSPF; therefore, any MTR-built forwarding table can be emulated by a set of policy-based route maps. This capability was used as an MTR surrogate on this program due to the unavailability of Cisco routers with MTR capability.

## **1.2 Methods, Assumptions, and Procedures**

This final report summarizes the contents of a number of previously published reports identified in Table 1-1. More technical details can be found in the respective deliverables. This document is organized as follows:

- Section 2 provides a concept of operations for MTR-RMA;
- Section 3 summarizes activities conducted as part of Task 1;
- Section 4 summarizes activities conducted as part of Task 2;
- Section 5 summarizes activities conducted as part of Task 3
- Section 6 summarizes activities conducted as part of Task 4;
- Section 7 concludes the main body of this report; and
- Appendix A provides a Memorandum documenting the transfer of the program software to SPAWAR Systems Center, San Diego.



In this simple example, multi-topology routing is described as having equivalent functionality as policy based routing and advertisement of static routes at the shore. However, except for simple scenarios, policy routing is cumbersome to configure and more brittle to topology changes. Multi-topology routing offers the potential for more flexible and fault tolerant design of alternate routing topologies that achieve desired load balancing goals even in the face of link failures, with topologies managed from within the routing protocol. In essence, MTR provides a dynamic, coordinated, distributed form of policy routing.

### 2.1.1 Components of MTR

MTR requires the deployment of three mechanisms in the network, and a management infrastructure to globally coordinate the operation. These mechanisms are straightforward extensions of the mechanisms already in use for the default (single) topology defined by legacy implementations.

#### 1) Dissemination of MTR metrics

In OSPF, link cost metrics are disseminated in Link State Advertisements (LSAs), which are flooded throughout the OSPF flooding domain. MTR metrics are similarly disseminated in LSAs.

For OSPFv2 for IPv4, the extension to provide MTR metric information in the LSAs is straightforward, since there are unused “Type of Service (TOS)” fields in the LSA formats (that were previously used for a similar Type of Service routing that was originally included in OSPFv2 but was later removed from the standard due to lack of deployment). MTR is slightly redefining the semantics of these TOS fields. For OSPFv3 for IPv6, the TOS fields do not exist in the LSAs, so separate LSAs must be generated to carry the multi-topology information. In general, where one router LSA was sufficient to carry default metric information, two router LSAs will be needed to carry the default and MTR metric information, although it may be possible in the future, when interoperability concerns are lessened, to stop circulating the legacy default LSAs.

#### 2) Computation of MTR topologies

MTR requires that routers compute shortest paths for each of the topologies active in the network. Routers must be modified to compute these topologies; in general, this will require running the shortest path computation code for each active topology each time that the shortest path computation is performed for the default topology.

Many routers are structured such that a routing information base (RIB) is maintained by the routing protocol process and a forwarding information base (FIB), or forwarding table, is used by the forwarding code, and the routing protocol(s) populate the FIB with routes. In MTR, both the RIB and FIB need to be extended to cover multiple topologies. In practice, this means that routers will have at least one forwarding table per routing topology.

#### 3) Mapping of packets to MTR topologies

At each forwarding hop, packets must be mapped based on some criteria to the appropriate forwarding table. This classification can be done based on a policy map in a Cisco router or packet classifier in a Linux-based router.

The MTR CONOPS document prepared under Task 1 of this contract goes into more details on MTR, including deployment considerations, a survey of alternatives, and the application of MTR to ADNS and IANM CONOPS [HKB+06]. Next, we briefly discuss its mapping to IANM and to coalition networks.

## 2.2 MTR and IANM

This section describes the relationship between MTR and Integrated Autonomous Network Management (IANM) Concept of Operations. From the IANM CONOPS draft [IAN05]:

The goal of the IANM is to transition Command and Control resource management from the relatively static and manual approach to a more Network-Centric automated framework, achieving the FORCEnet objectives and providing a capability on demand in a responsive manner. IANM provides the *Warfighter (afloat communications planner and watch officer)* the ability to dynamically allocate and prioritize *network* and *communications resources* based on the *Operational Commander's* intent. During operations IANM will inform the warfighter of communications performance relative to mission tasks, and will offer clear, concise, courses of action should task performance degrade to unacceptable levels.

To accomplish this goal IANM will configure, manage, monitor, and assess the ship-to-ship and ship-to-shore Internet Packet (IP) communication resources. The IANM will advance Naval communications resource management and the communications planning process from the current relatively static and manually intensive process of today to an automated, dynamically responsive capability that will improve Fleet communication capabilities and take full advantage of advanced Internet Protocol (IP) technology.

MTR is one low-level tool that an overall IANM system could use to improve IP routing based on the communications plan in effect. Figure 2-2 (from [IAN05]) depicts the relationship between IANM users and functions of the network. In particular, the IANM CONOPS describes the roles of a number of different users of IANM, ranging from Mission Task Planners working to define the OPLANS to support the required OPORDS. For example, if the OPORD is “provide security for an area at sea”, an OPLAN might contain tasking to “conduct air surveillance every 10 minutes” [IAN05]. While MTR may be used to support such an OPLAN, it would not typically be visible to an IANM user operating at this level of description of network capabilities.

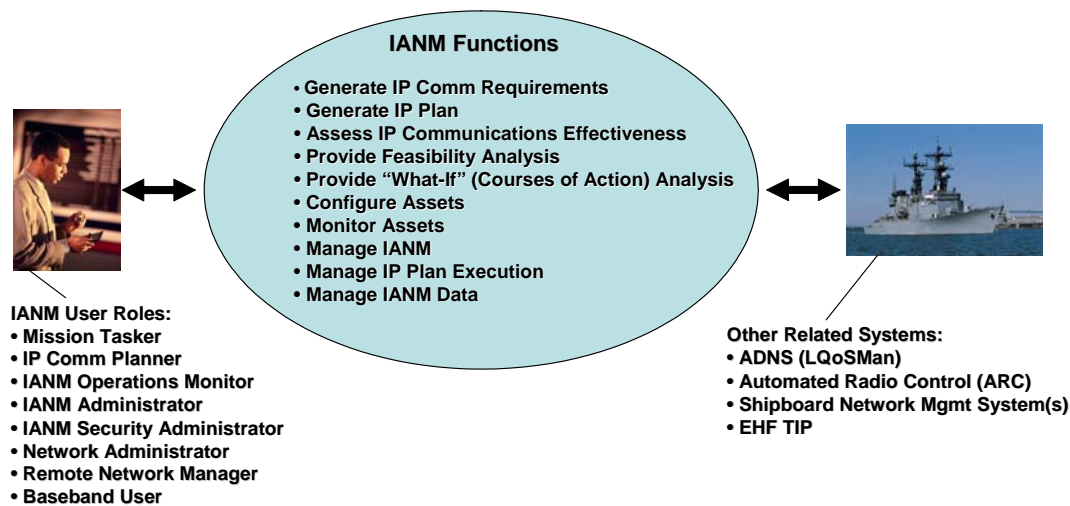


Figure 2-2. IANM Users and Functions

The next level of granularity in mission planning is the Strike Group Communications Planner, who takes the OPLAN and generates an OPTASK COMM plan or file that includes the radio configurations, including frequencies, emission codes, and crypto keys, and associated host applications for both “circuit-oriented” and “IP-oriented” services, along with platform responsibilities. Since this level of planning coordinates across multiple platforms, it is likely at this level that MTR functionality is expressly considered by the planner. For example, a plan may require that a particular application be given low latency and high priority through the network; the communications plan should outline the MTR topology and QoS configuration necessary to effect the plan. It is anticipated that a GUI-based tool will be provided to such a planner, rather than relying on low-level configuration syntax, and that IANM will provide such a GUI-based tool.

Finally, the Platform Communications Planner and IANM Operations Monitors are responsible for implementing the plan on a per-platform basis and monitoring its effectiveness, respectively. Operations at this level likely require an understanding of MTR configuration and its relationship to the overall communications plan, and a monitoring capability to assess the performance of the current configuration. The latter capability (assessment) is also a functional component of the IANM program.

IANM is envisaged to provide a unified management and human-computer interface (HCI) framework for disparate network management components in the present ADNS, such as Automated Radio Controller (ARC), LQoSMan, The Assessment Profiling System (TAPS), and commercial network management platforms. MTR configuration and management is one element of this framework that touches upon the core planning, analytical and assessment, and visualization capabilities that IANM provides.

In Sections 3 and 4 below, we summarize how MTR was mapped to the IANM system from an implementation perspective.

### 2.3 Policy-Based Routing as an MTR Approximation

The class of Cisco routers considered for Trident Warrior 2008 and ADNS Increment III (Cisco 2800- and 3800-series routers) was expected to support MTR at some point during the contract. However, as of the writing of this report, such implementations have not been released. In 2006, we were recommended by ONR and SPAWAR to explore an alternative design based on the selective application of policy-based route maps to respond to operator policy rules, dynamic congestion, and topology changes. We call this Multi-Topology Routing via Route Maps (MTR-RM) as it represents a surrogate MTR capability.

Policy-based routing (PBR) is a technique used to override destination-based forwarding tables. Unlike destination-based forwarding, PBR can make forwarding decisions based on other factors, including source address. It can therefore be used to implement class-based routing or source-based routing (where “policy-based” routing is the general term). PBR is implemented by building a route map against which packets are first matched, during the forwarding process. If there is a match, the packet is directly forwarded to the interface specified in the route map, without consulting the forwarding table. If there is no match, the packet is forwarded based on the traditional forwarding table. A few limitations to note about PBR are that (i) these PBR route maps are statically configured, and (ii) inconsistent policy configuration can cause forwarding loops.

For IOS-based Cisco OSPF routers not implementing multi-topology routing, policy-based routing may be used to achieve similar effects. To use policy-based routing to achieve similar effects it is necessary to create topology abstractions that are overlaid on the routing core. In other words, for each orthogonal topology, it is necessary to identify a set of routers and links belonging to the given topology. One or more classes of traffic are then associated with each topology. In the IANM project, we have therefore written our MTR Analysis modules to generate information used for policy-based route maps that approximate MTR routing configuration.

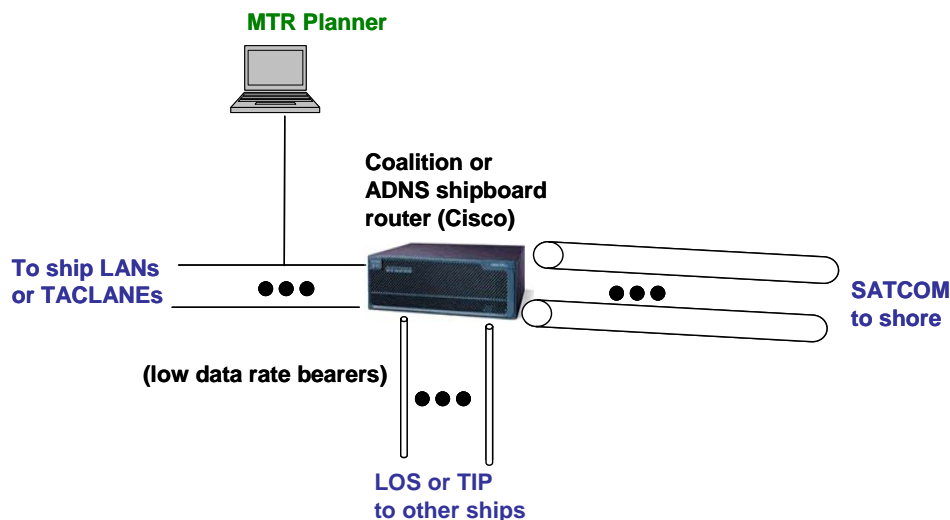


Figure 2-3. Distribution of MTR functional elements in the IANM implementation framework.

The concept of operations when using MTR analysis via route maps is similar to that of using MTR. The main difference is that route maps are uploaded instead of MTR link metrics, and such route maps must be evaluated for robustness to topology faults, whereas MTR routers let the OSPF shortest path calculation take care of topology faults (in other words, using route maps puts more of a burden on the planning tool that configures them). For example, Figure 2-3 shows a notional “MTR Planner” that is connected to a Cisco router on a link residing in the same security enclave as the router. The router, in general, is assumed to run OSPFv2, and has three groupings of interfaces:

1) SATCOM links to shore; these may be managed by Cisco Optimized Edge Routing (OER) (a Cisco-proprietary technique for managing static or BGP-learned routes at edge routers) or use static policy-based route maps, and may be in a separate area than the other interfaces;

2) line of sight (LOS) or other (EHF TIP) links to other ships; and

3) Internal LAN segments, connected to SECRET router or TACLANEs (if an ADNS router).

If the router is not the ADNS router but is instead a separate coalition router, the same conceptual diagram applies, except that the links labelled “SATCOM to shore” are more likely a single link to a TACLANE for tunneling over the ADNS network.

The operational problem is to make effective use of the LOS links by spreading application traffic appropriately. The technical problem is that OSPFv2 does not allow load-balancing across the links unless they are configured to be of equal cost, and equal-cost multipath does not work across multiple areas (both ADNS and Coalition network configurations are planned to place LOS links in a different OSPF area than the Satcom link). Therefore, solutions within the existing OSPFv2 framework alone are limited. The MTR Planner software developed in Task 4 of this contract was designed to help manage exactly this scenario [Hen08].

### 3. Task 1 - System Engineering

The goals of this initial task were to define the systems requirements on the MTR component of IANM, to describe how MTR management related to the Navy and IANM CONOPS, to define a software architecture, and to deploy a laboratory and development testbed.

The main documents produced by Boeing during this phase were:

- MTR Software Design document [BHK06b]
- MTR CONOPS document [HKB+06] and presentation
- Spiral 1.1 Demonstration Plan document

all of which were uploaded to the IANM portal.

The CONOPS development has been previously summarized in Section 2 above. Here, we briefly summarize the software architecture work and the early software development work that led to the first demonstration in March 2006.

#### 3.1 Software Architecture

The Software Design Document [BHK06b] is a comprehensive report of the software design and development through June 2006. The document identifies requirements on and revised the CONOPS for use of MTR within IANM and ADNS, and covered the overall software architecture, identified key functional modules, defined interfaces between those modules, and described logic used by the analytical portion of the software. The document also described Boeing's development testbed and demonstration results and future plans. Here, we provide a brief overview of the software design.

Our overall software architecture was built to be compatible with the IANM framework. Figure 3-1 illustrates the main functional components and their relationship to other IANM components. From an implementation standpoint, the MTR Data Collection Engine (DCE) was implemented as a Java-based XCAFE plug-in (for SFA's XCAFE environment). The MTR Planning Tool was assumed to be implemented on the same IANM server as the IANM Communications Enterprise Planner (CEP). Two elements of MTR DCE ("MTR Agent" that gathers current routing configuration from routers, and "NetFlow Collector" that gathers samples of the traffic matrix) were implemented as XCAFE plugins, and published their gathered data using Java Message Service (JMS) messaging to publish collected and aggregated data to the IANM data repositories. The MTR Planning Tool was functionally part of the Communications Enterprise Planner (CEP), responsible for distilling commander's intent to an IP communications plan as part of the Comms Planning Tool. This planning tool can use either real-time estimates of traffic flows and network state or canned estimates when used in mission planning mode. The database schema and interface was implemented using tools (AndroMDA, Magicdraw) to generate Java-based object shells to wrap the relational (MySQL) database. Finally, the C4ISR Enterprise Manager (CEM) provided overall network management and visualization, and MTR was expected to support this overall framework. The MTR Planning Tool interfaced with external components as a JMS message-driven bean in the JBoss environment. Algorithms used by the planning tool were implemented in C/C++ and were callable through the message-driven bean implemented in Java and described above. The description of the IANM implementation of all components other than MTR Planning Tool and MTR DCE is outside of the scope of this document.



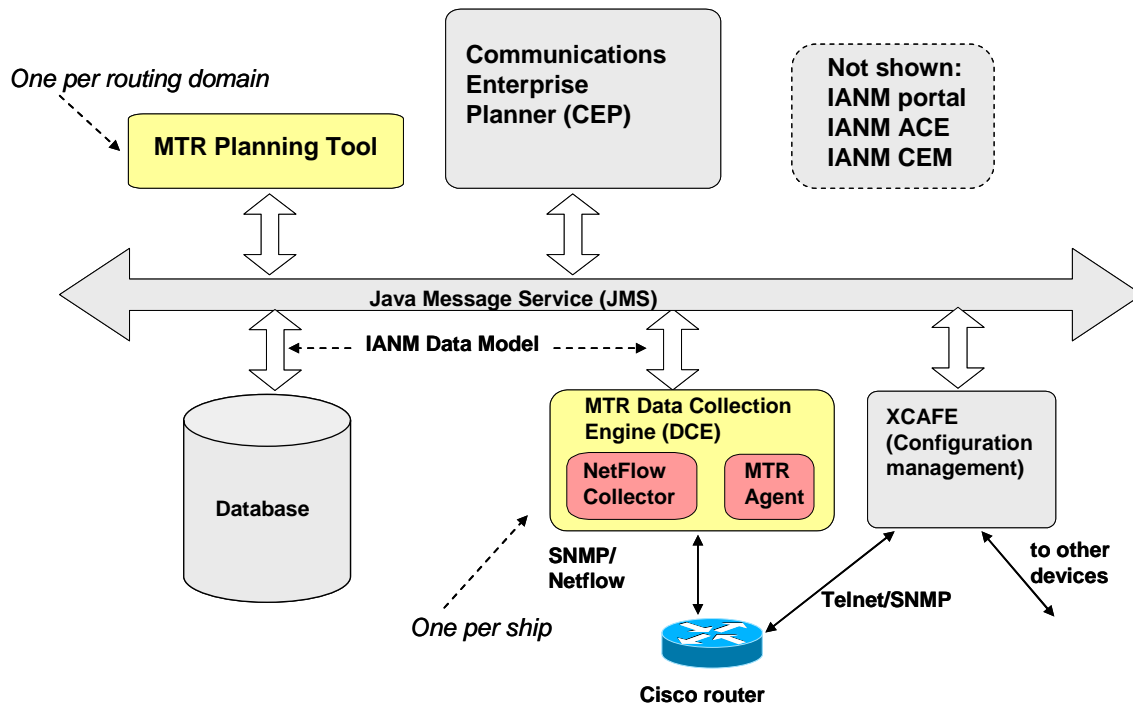


Figure 3-1. Distribution of MTR functional elements in the IANM implementation framework.

Software elements in IANM interact on the basis of queries and responses to a common Data Model in a JBoss environment. For instance, a request to generate a MTR routing plan is formulated by creating an object named *MtrRoutingPlanRequest* in the database, and by sending the MTR Planning Tool a handle to an instance of this object. The *MtrRoutingPlanRequest* object identifies which type of routing optimization is to be performed, and provides handles to all of the input data (stored in other objects in the database) needed by the MTR Planning Tool to execute the optimization, including the traffic matrix, OSPF network graph, additional configuration information on the routers, configuration information on the MTR class mapping, and routing policy information. The MTR Planning Tool then computes a routing plan for MTR containing the MTR routing configuration information and stores it in the database as a *MtrRoutingPlan* object, where it may be fetched by other IANM elements (for example, to be transformed into a text-based Cisco IOS routing configuration and pushed to the routers). Finally, it sends the handle of the *MtrRoutingPlan* object instance to the originator of the *MtrRoutingPlanRequest* (in IANM, the CEP).

We have provided the MTR-related schema database in the form of MagicDraw Data Model objects, and the Data Collection Engine (DCE) and Planning Tool components. Below, we briefly summarize the software strategy for each component.

### 3.1.1 MTR Data Collection Engine

The MTR DCE was implemented as an XCAFE plugin. Written in Java, the DCE interacts with XCAFE via XCAFE's service interface. The MTR Agent requires a bootstrap configuration file (stored in XCAFE) to identify the IP management interface on each Cisco router under management. Similarly, the Netflow Collector requires a configuration of the router IP addresses and port numbers on which to listen for NetFlow PDUs. Data is ultimately written to the database via the XCAFE service interface, which publishes data to a JMS "topic" (roughly analogous to a multicast destination address).

### 3.1.2 MTR Planning Tool

The MTR Planning Tool was implemented in C++ and was wrapped as a shared object callable from Java through the ubiquitous Java Native Interface (JNI). The planning tool was intended to be run as a message-driven bean in the JBoss operating environment, listening for messaging on a conceptual queue from the CEP or other IANM entities, but ultimately the tool was driven from manual operator intervention since the higher-level CEP agents were never developed or exported an interface to the planning tool.

## 3.2 Software Development

In addition to defining CONOPS and the software architecture, we accelerated our software development schedule to meet the integration goals for the series of IANM demonstrations for 2006. The first integration-based demo was held in early March, 2006. At that point in time, we had accomplished the following software milestones:

- Defined a baseline MTR data model (version 1.1)
- IANM laboratory testbed deployed at Boeing, with IANM server and Cisco routers
- Baseline Data Collection Engine (DCE) implemented
- Baseline DCE/database interface implemented
- The DCE harvested live data from network components and populated the IANM repository
- The MTR DCE was integrated into the XCAFE framework in collaboration with SFA
- Software for implementing the MTR Planning Tool had been initiated

The Spiral 1.0b demonstration in March 2006 centered on illustrating the capabilities of the MTR Data Collection Engine (DCE), summarized below in Section 4. The NetFlow collector and MTR agent collected the network traffic and the topology data from a set of Cisco routers and wrote the data, formatted according the data model schema, to the database. The demonstration also showed the real time reaction of the software to the dynamic topology and traffic changes in the network. Figure 3-2 depicts the hardware configuration of the demonstration platform.

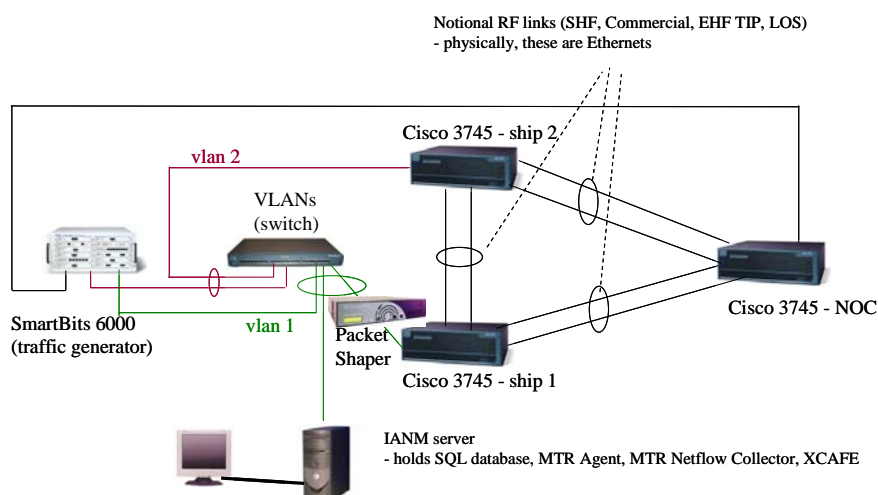


Figure 3-2. Hardware configuration for Spiral 1.0b demonstration of DCE software

The version of DCE software demonstrated at the Spiral 1.0b demonstration was integrated into the SFA's XCAFE framework as shown in Figure 3-3. Each DCE component had been converted into XCAFE plug-ins.

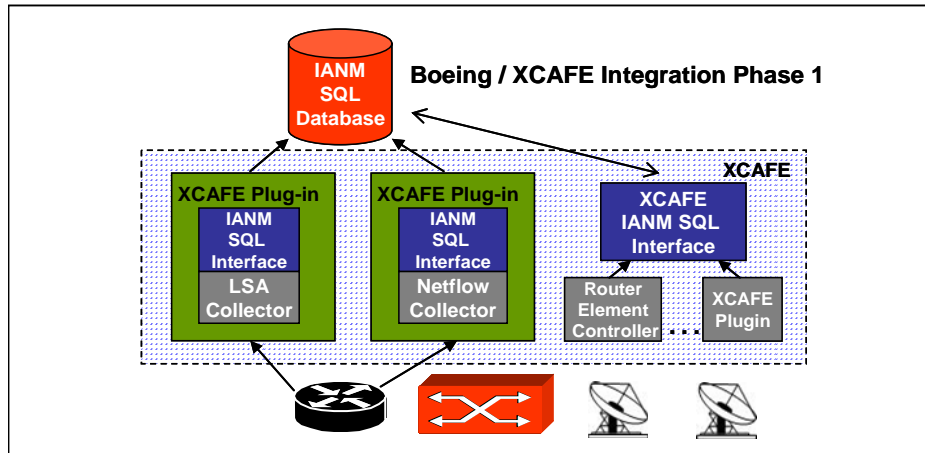


Figure 3-3. MTR DCE and XCAFE Spiral 1.0b Demo Integration

## 4. Task 2 - MTR Data Collection Engine

The first official software deliverable (CLIN 0002) for this contract was the MTR Data Collection Engine (DCE) in September of 2006 [BHK06]. The MTR Data Collection Engine (DCE) software provides two main functions on networks containing Cisco routers: dynamically collecting network topology and network traffic load information. We summarize herein the MTR DCE deliverable that has been more completely documented (such as a detailed installation and usage guide, and testing results) in [BHK06]. The software was also delivered under CLIN 0002.

The MTR Data Collection Engine was functionally divided into a MTR Agent component and a NetFlow Collector component, both introduced above in Figure 3-1.

### 4.1 MTR Agent

The MTR Agent interfaces with a Cisco router via the Simple Network Management Protocol version 2c (SNMPv2c)(which uses the services of UDP to the SNMP agent residing on the Cisco router). Here, we summarize how the various MTR Agent processes, contained within the DCE, interact with a Cisco router to collect router state and how they interface with other components in the broader IANM system.

The MTR Agent is implemented in the Java programming language within the IANM framework as an XCAFE plug-in. It consists of process threads started by the XCAFE subsystem and runs under the control of the JBOSS application server. It retrieves SNMP MIB information from a set of Cisco routers specified in a configuration file. There are two steps to bootstrap the Cisco router and MTR Agent. Each Cisco router is configured (via CLI commands) to enable SNMP traps and with the IP address of the MTR Agent, which acts as a trap collector for traps emitted by a given router. The routers under the MTR Agent's management are specified in its configuration file and are identified by their respective IP management interface addresses.

The MTR Agent, upon startup, polls the first Cisco router on its configuration list for every OSPF link state database entry contained within the router's OSPF database table (a dump of the entire routing domain routing database), and then polls each router on its configuration list for objects contained in the interface and IP address entries. The data model objects shown below directly correspond to well-defined SNMP MIB objects. The MTR Agent publishes this information to the database as an *MtrRouter* object; an *MtrRouter* object represents a managed router; its subordinate objects represent well known SNMP managed information base (MIB) tables. Subsequently, the MTR Agent polls the routers for updated MIB objects at timed intervals or upon the receipt of an SNMP trap from a device under its management that indicates a change in the router state that may affect routing. Whenever the MTR Agent detects a change in router state it publishes a new set of *MtrRouter* objects. The *MtrRouter* objects are stored in the database, indexed by timestamp, for later use by the MTR Planning Tool.

The MTR Agent was implemented as two threads of concurrent execution. The first asynchronously monitors a UDP socket for the receipt of valid SNMP Traps. The second asynchronously schedules polling events that control the collection of router state using SNMP Get-Next PDUs.

### 4.2 NetFlow Collector

The NetFlow Collector is responsible for collecting information from Cisco routers about the active traffic flows and distilling this information into an *MtrTrafficMatrix* object, published to the database. NetFlow is configured on each router to emit NetFlow PDUs towards the NetFlow Collector's IP address and port. In the IANM demonstrations, there was one NetFlow Collector, but in an actual deployment, there would likely be one per ship. When the NetFlow Collector listens to more than one Cisco router, it must do so on different ports. Cisco NetFlow defines a flow as a unidirectional stream of packets identified by the following seven key fields: source IP address, destination IP address, source port number, destination port number, layer 3 protocol type, the input logical interface (ifIndex) and the TOS byte. A combination of these fields defines a unique flow.

To generate the optimal network configuration, the MTR Planner algorithm requires the current physical topology the network and the respective traffic matrix. The records of individual traffic flows can be pre-populated by the IANM CEP, but can also be generated in real-time by Cisco NetFlow. The NetFlow Collector module collects, aggregates, and publishes the flow records to the database.

Figure 4-1 illustrates some detail on this process. Cisco routers at the boundary of the MTR routing region are configured to monitor only the traffic entering the MTR routing region. The matrix generation algorithm

periodically retrieves the flow records stored in the database and processes them in an incremental time window. The retrieved data is processed to associate the represented flows to the respective ingress and egress routers. Consider that subnets A and B in Figure 4-1 represent enclaves, and that Router C is one such MTR edge router. Assume that the top host in the diagram generates 100 units of traffic from network A to network Z using DSCP codepoint 0 (the host could mark its own traffic or a Packetshaper, not shown, could be used for traffic marking). Assume also that another host in subnet A generates 200 units of flow to the same destination, but on a different DSCP, and that a host on subnet B generates 500 units of flow to subnet Z. From the perspective of MTR routing configuration, the individual source subnets are not necessarily needed by the MTR Planning Tool and the red and green flow can be aggregated into a single flow of 600 units; Cisco NetFlow supports such aggregation. On the other hand, when using MTR emulation via policy-based route maps, the source subnet prefix can be used for providing further granularity, and the Cisco NetFlow could be configured to track each flow individually.

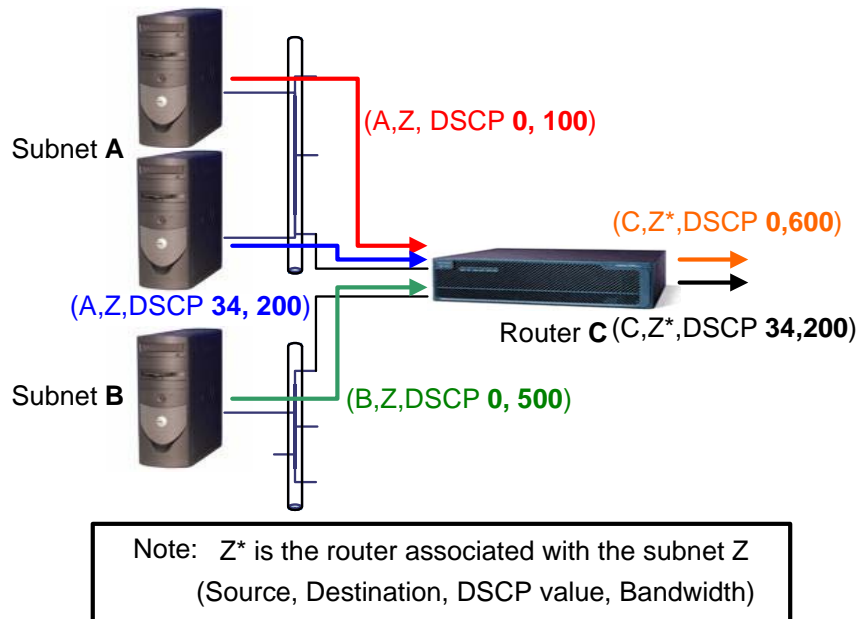


Figure 4-1. NetFlow Collection.

The implementation of the Netflow Collector relies on the build environment that XCAFE plug-in generator provides; the underlying source code resides in the lowest layer directory. The implementation is in Java, and configuration is done based on an XML file. More details are in the documentation for CLIN 0002.

## 5. Task 3 - MTR Basic Analysis

The MTR Basic Analysis module was delivered under CLIN 0003 of the contract, in March 2007, although initial aspects of this module were demonstrated a year earlier in the Spiral 1.0b demonstration in March 2006. The MTR Basic Analysis software takes as inputs the desired communications plan in the form of a detailed traffic demand matrix, and produces a recommended routing plan. As such, it complements the Data Collection Engine (DCE) described in the previous section, whose job it is to build the traffic matrix. Here, we summarize the MTR Basic Analysis component; a detailed description of this component, including the functional architecture of the software, its relation to the overall IANM project, installation instructions, and a summary of test results, is contained in the documentation for the CLIN 0003 software [BHK07]. The MTR Basic Analysis software updated and extended the software delivered under CLIN 0002.

In addition, Boeing demonstrated the MTR software at a demonstration at SPAWAR Systems Center, San Diego, in February 2007. Some of the below material is summarized from that demonstration. After this demonstration, Boeing began to develop the distributed implementation, decoupled from other IANM software, that became the CLIN 0004 deliverable in February 2008.

### 5.1 System Software Flow

The following set of diagrams illustrates the functional flow of our software and the MTR Planning Tool. The diagrams are taken from a small-scale demonstration conducted on 28 February 2007 at the SPAWAR SSC-SD Building 40 CNDL Laboratory, notionally involving three ships and a NOC (Figure 5-1). The corresponding hardware configuration is shown in Figure 5-2.

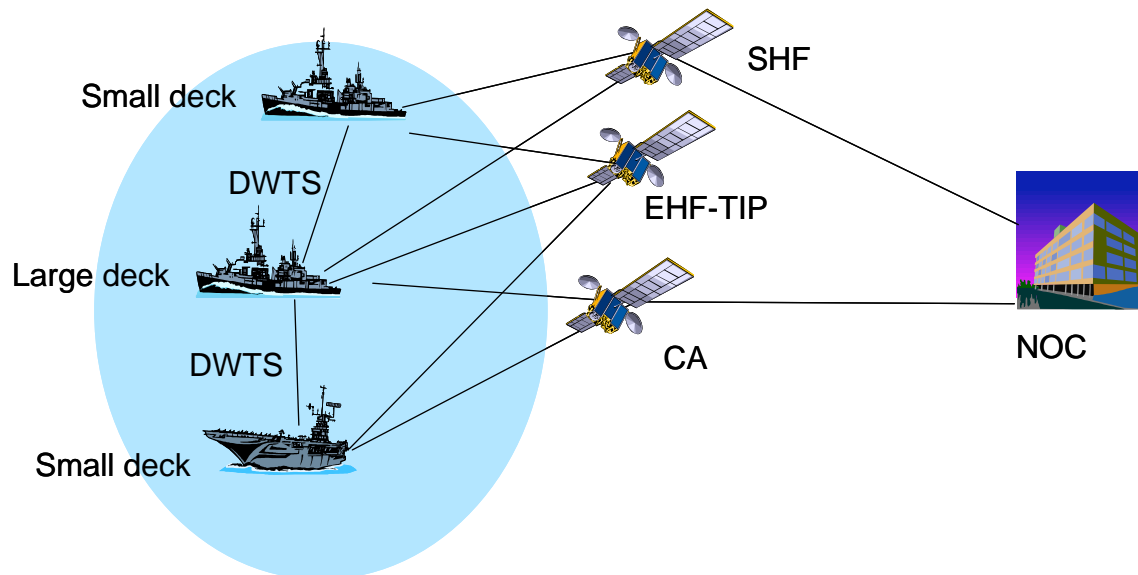


Figure 5-1. Notional view of system, for software flow description.

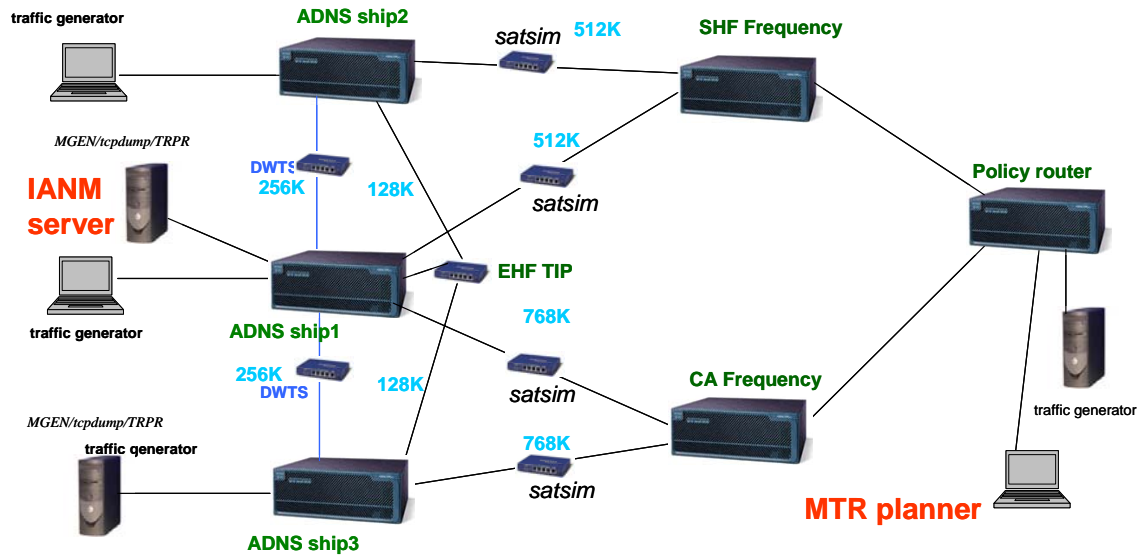


Figure 5-2. Hardware representation of the system in Figure 5-1.

Figure 5-3 provides a simplified view of how a traffic matrix may be built by an IANM server or by several IANM servers (not shown). The basis for the MTR Data Collection Engine (DCE) is the use of Cisco NetFlow, which reports the presence of traffic flows, matched to a certain criteria, to the NetFlow collector. In an actual deployment, there would be an IANM server on each ship, and data would be published to an IANM repository, where it could be assembled by another agent to form the traffic matrix. Here, we assume that the IANM server is able to publish the traffic matrix to the MTR planner (Figure 5-4).

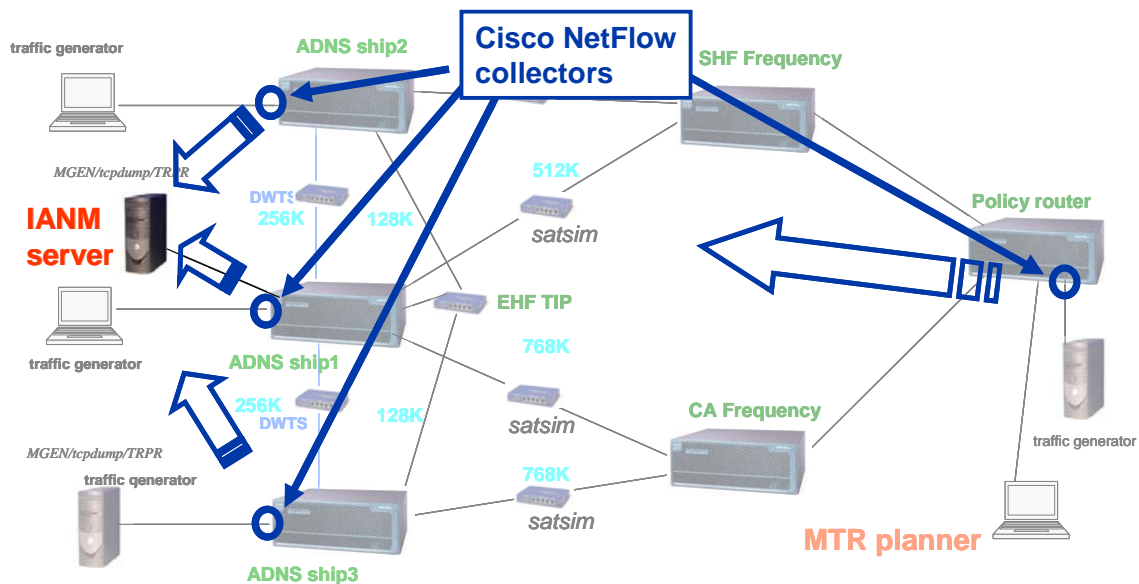


Figure 5-3. Gathering of raw data to support traffic matrix generation

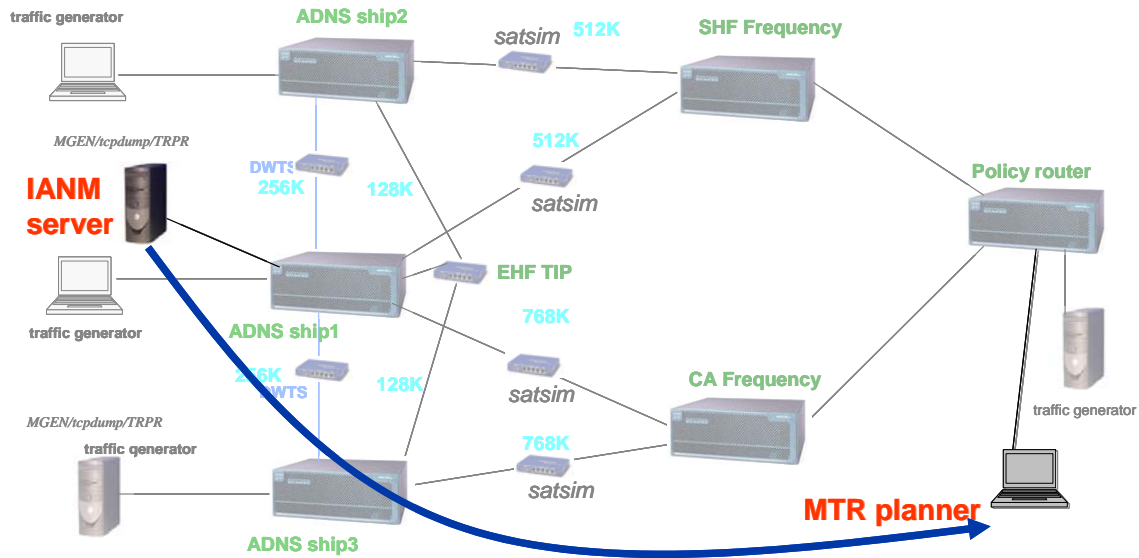


Figure 5-4. Publishing of the traffic matrix to the MTR Planner.

The MTR planner needs additional information besides the traffic matrix. It also needs the OSPF database for the topology, interface configuration information from the routers in the topology, and ancillary information about mappings of DSCP values to topology, and relative priorities between flows and classes of service. Figure 5-5 shows how the information can be gathered directly from the routers using SNMP (solid arrows) and via passive OSPF adjacency forming with a nearby router (dotted arrow).

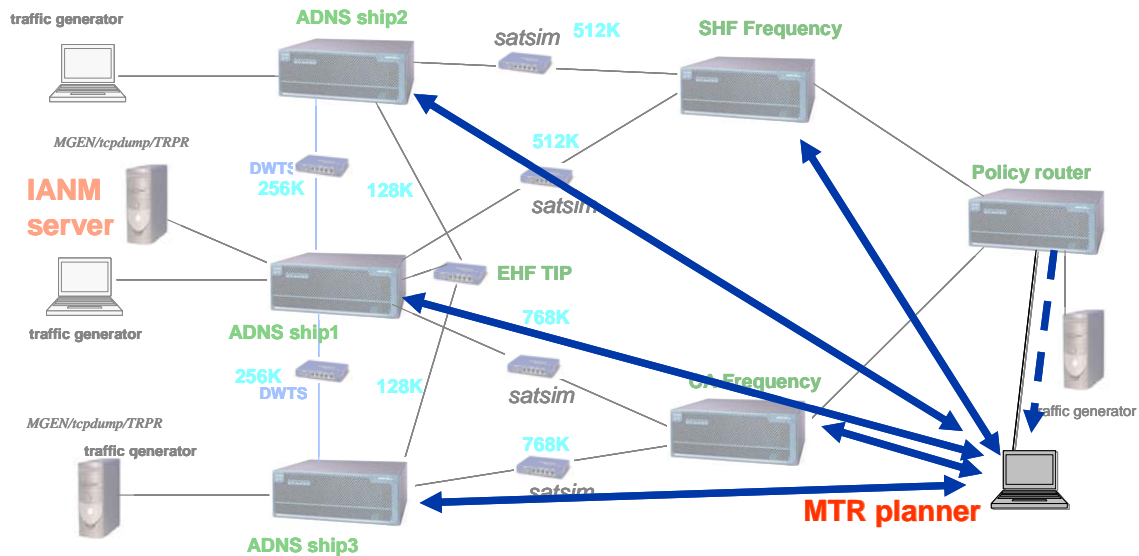


Figure 5-5. MTR planner gathers interface information (solid arrows) and topology information (dotted arrow).

The MTR planner next generates a routing plan according to the desired objectives (such as load balancing), based on the traffic matrix, OSPF network topology, router interface information, and information regarding flow priorities. This routing configuration is published to a database or alternately pushed directly to the routers (Figure 5-6).



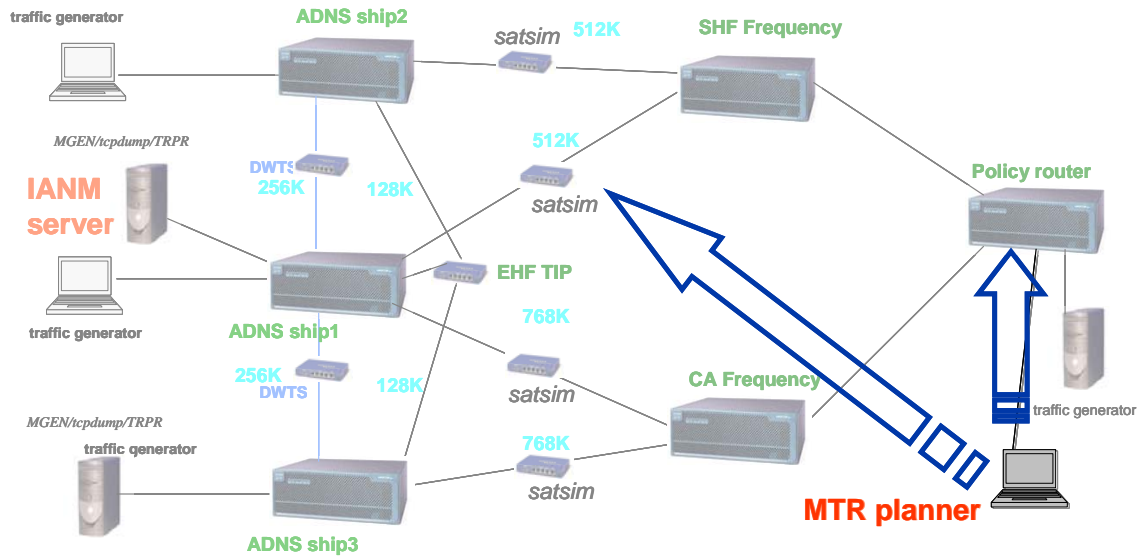


Figure 5-6. Publishing of MTR routing configuration to the routers.

MTR planning software that performed the above operations (Figure 5-5 and 5-6) was demonstrated on 28 February 2007 at the SPAWAR SSC-SD Building 40 CNDL Laboratory. In addition, the system was demonstrated to be robust to link failure (MTR planner detected the loss of link and was able to replan the traffic balance) and change in traffic matrix (again resulting in a new plan).

## 5.2 Development of a Routing Plan

Figure 5-7 illustrates the general flow for developing a routing plan. This section provides an overview into the technique used by the MTR Planner for generating such a plan.

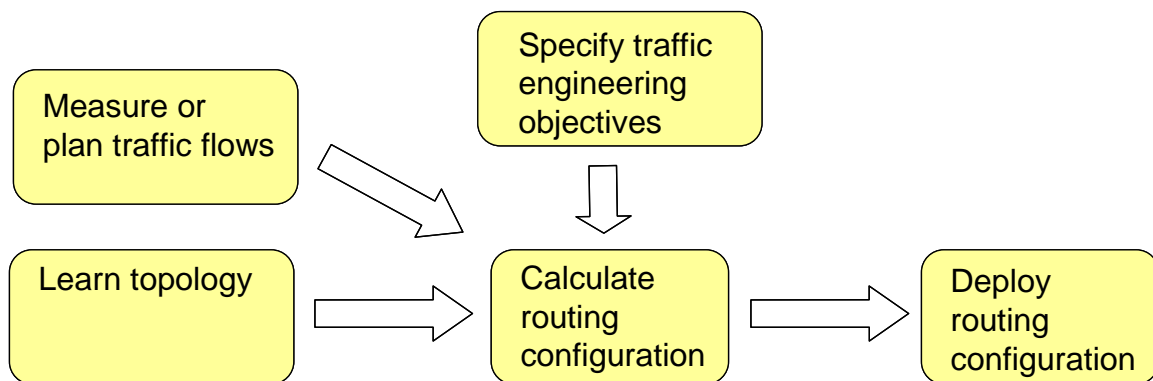


Figure 5-7. Flow description for development of a routing plan

### 5.2.1 Inputs

The following inputs are used in calculating a routing plan.

**Topology.** A full OSPF link state database (LSDB) provides a complete network map of the connectivity and default link weights of the network topology.

**Interface information.** The OSPF LSDB is supplemented with information about the link capacities of each router interface and the capacities of topologies configured on each interface; such information is not stored in the OSPF link state advertisements. Additionally, there may be quality-of-service (QoS) information on each interface that allocates a portion of link capacity in the queues to different classes of service; this information can be used in the route mapping computation. Interface IP addressing information is also used in the generation of route maps for

numbered interfaces.

**Traffic matrix.** A set of flows identified by the following tuple: (ingress subnet, egress subnet, DSCP value, capacity). If the flow records are provided in terms of IP addresses, an additional mapping must be performed to obtain the ingress and egress router interfaces. These ingress and egress router interfaces are independent of the routing that occurs within the MTR-managed routing region (i.e., they are invariant so that there is no cyclic dependency).

**Traffic engineering objectives.** A definition of the objective to satisfy. This may be a simply defined objective such as “load balancing” or a more sophisticated objective such as fine-grained policies about which types of flows can traverse which network paths.

**DSCP to MT-ID mapping.** A mapping of one or more DSCP values to a multi-topology ID value.

### 5.2.2 Outputs

The MTR planner can be used to output either a set of link weights or a set of corresponding route maps; this iteration of the software generates a set of route maps as its output.

**Link weights.** This would specify, for each MT-ID, a link weight for each MTR-capable interface in the system.

**Route maps.** This would specify, for each router, a set of route map entries that match the following tuple: (source IP subnet, destination IP subnet, topology identifier) to an outgoing interface; access list rules that associate classes of traffic to route maps.

### 5.2.3 Calculation

The generation of a network-wide set of OSPF link metrics to achieve optimal traffic flow is known to be a problem of NP complexity; this is because every permutation of the graph representing the network must be enumerated and compared in order to find the set that achieves the objective. Because of this, it's desirable to employ a heuristic algorithm, ideally with low-order polynomial complexity to keep computing resource utilization as low as possible.

We have implemented a heuristic algorithm that achieves low-order polynomial complexity. The algorithm is iterative in nature; it finds a path through the network meeting the desired characteristics of each flow represented within a traffic matrix. It uses Dijkstra's algorithm to find a path; once a path is found it reduces the available bandwidth on each graph edge on the path and moves on to the next flow. Since Dijkstra's algorithm, of complexity  $O(n^2)$ , is run once per-flow, the complexity of the algorithm is  $O(mn^2)$ , where  $m$  is the size of the traffic matrix measured by number of flows. In our implementation, the traffic matrix has flow entries aggregated per subnet. The software implementation of the algorithm is restricted to use with single-area OSPF networks. The algorithm generates partial solutions in the case that the algorithm can't find paths through the network meeting the characteristics of some subset of flows in the traffic matrix.

More details of this algorithm (in pseudocode) can be found in the CLIN 0003 documentation

### 5.2.4 Generating MTR Link Metrics

The MTR weight generation algorithm was easily modified to generate MTR Link metrics. The principal changes to the algorithm relate to the adjustment of the link capacities and link selection.

**Adjustment of link capacities:** In this variant of the algorithm, the link capacities are adjusted only after all of the flows for a topology are mapped, rather than after each flow is mapped; the capacity consumed by the topology is removed from the total capacity of the link.

**Link selection:** Once an outbound link attaching one router to another router is included in a topology, it must be used for all flows that transit those two routers. Dijkstra's algorithm is modified to enforce this restriction.

**MTR weight assignment:** After all flows have been mapped, metrics are assigned for each interface on each participating router. For a given topology on a given router, each outbound interface on the router, for the given topology is assigned a weight of 1; the weight for each remaining interface, for the given

topology, is set to the maximum network diameter + the inverse of the link's residual capacity.

### 5.2.5 Implementation

This section summarizes the interfaces and implementation details on the MTR Basic Analysis component, also colloquially referred to as the MTR Planner. The MTR Planner has interfaces to several other components in an IANM system. Figure 5-8 summarizes the interfaces described in this section.

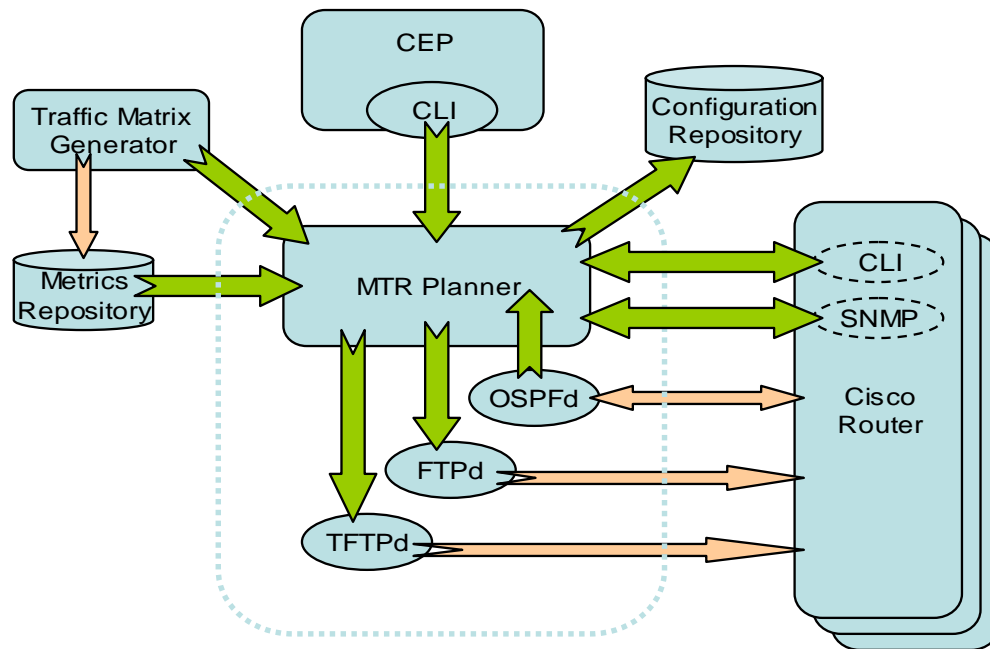


Figure 5-8. MTR Planner interfaces.

**Interface to Communications Enterprise Planner:** The MTR Planner was designed to be invoked as a service by an operator using a command-line interface through the Communications Enterprise Planner. The MTR Planner CLI consists of a main menu and three sub-menus in which related functions are grouped. The sub-menus are accessed from the main menu. The three sub-menus are the *Router menu*, the *Options menu*, and the *Debug menu*. The *Router menu* is used to build and manage router configurations; the *Options menu* is used to manage MTR Planner options controlling its behavior; the *Debug menu* is used to enable various MTR Planner debugging controls.

**Interface to Metrics Repository:** The Metrics Repository is a conceptual database containing mission parameters, network state, and real-time metrics in support of a mission. As of this deliverable, the IANM project had not defined a separate Metrics Repository; there was only a single IANM Data Model. Under command from the CLI, the MTR Planner retrieves the most recent Traffic Matrix stored in the Metrics Repository. The flow records are parsed and stored within internal data structures. The source code implementing the interface to the metrics repository is contained within the `mtrPlanner.C` source code file.

**Interface to Traffic Matrix Generator:** The MTR Planner has an interface to the Traffic Matrix Generator that serves as an alternate source of Traffic Matrices. When the MTR Planner is built to operate in a real-time monitoring mode it would normally use this interface. When this interface is used, the Traffic Matrix Generator asynchronously sends traffic matrices to the MTR Planner over a TCP stream socket using a message format defined for this program.

**Interface to Cisco routers:** The MTR Planner interfaces directly to a Cisco router under management through:

- The router's telnet Command Line Interface (CLI)

- The SNMP agent residing on the router under management.

The telnet CLI is used to instruct the router to use tftp or ftp to transfer a configuration file that is applied to the running configuration using the IOS CLI “copy ... running-config” and to retrieve the status of this operation.

The MTR Planner exchanges SNMP GetBulk PDUs with the SNMP agent residing on the managed device in order to retrieve state from the router.

**Other interfaces:** The MTR Planner interfaces to the Quagga OSPFv2 daemon to monitor and dynamically receive updates to the OSPF Link State Database for the OSPF routed network under management; it interfaces with an ftp or tftp daemon to transfer configuration files that it builds to Cisco routers under management.

## 6. Task 4 - MTR Analysis with Policy Constraints

During the course of the program, it became evident that Cisco routers of the type that the Navy uses at sea were not going to implement MTR in the timeframe of the contract. That precipitated a change to using policy-based route maps as a surrogate for MTR, as already described above. In 2007, we received additional feedback that led to Boeing's MTR software being decoupled from the rest of the IANM software. First, the Trident Warrior 2007 exercise did not involve multiple ships, and the experimenters were not permitted to touch the router configuration anyway. This led to a separate Boeing demonstration (mentioned above in Section 6) in February 2007.

Some additional feedback from PMW-160 attendees at the August 2006 demonstration also suggested some further decoupling. In particular, Boeing was told that ADNS Increment III would base its load balancing solution for ship to shore traffic on a Cisco technique called Optimized Edge Routing (OER). Demonstration attendees also voiced a suggestion that MTR management (or any router management) could be thought of as a tool for providing (implementing) service level agreements, and that the role of the rest of the IANM would be to generate these service level agreements based on commander's intent. Issues regarding traversal of security boundaries were also raised. In the end, we agreed with SPAWAR that, for the remainder of the project, MTR and IANM was a secondary goal and is not strictly required, and that if decoupling the software would improve its chance for transition or consideration by PMW-160, then that was the approved course of action.

As a result, we performed this decoupling, and we did some testing of the Cisco OER technology. We also found that the Allied Multi Bearer Tactical Network team that had experimented with MTR in Trident Warrior 2006 was interested in any follow-on capability that could help to manage outbound links from a Cisco router. After several discussions, we oriented our final software deliverable towards the AMBTN networking scenario, and focused on participating in a Limited Objective Experiment (LOE) in January 2008. We tested and demonstrated our final software deliverable (CLIN 0004, summarized herein) and transferred it to SPAWAR Systems Center, San Diego. A memorandum regarding this transfer is attached as Appendix A to this report.

In summary, our final software deliverable [Hen08] worked as follows:

- 1) the MTR Analysis module is a set of Linux-based software processes that use policy-based route maps to implement multi-topology routing functionality;
- 2) an instance of this software runs on each ship and manages the outbound bandwidth from each ship;
- 3) the software works with Cisco routers presently being deployed as part of ADNS Increment III and other shipboard deployments such as Centrix;
- 4) the policies governing the rerouting of traffic are specified by the network operator in a data file, defining which applications preferably flow over which link when the default path available by routing is congested. Additional policy inputs allow the operator to define which flows are rerouted first in the case where multiple such flows are candidates for rerouting.
- 5) the optimized topology generated consists of additional policy-based route maps with higher preference than the default policy-based route maps that may have been implemented.
- 6) the reliance on IANM-provided software components was eliminated; all software runs on a Linux machine as either a C++ or Java process.

## 6.1 Goals

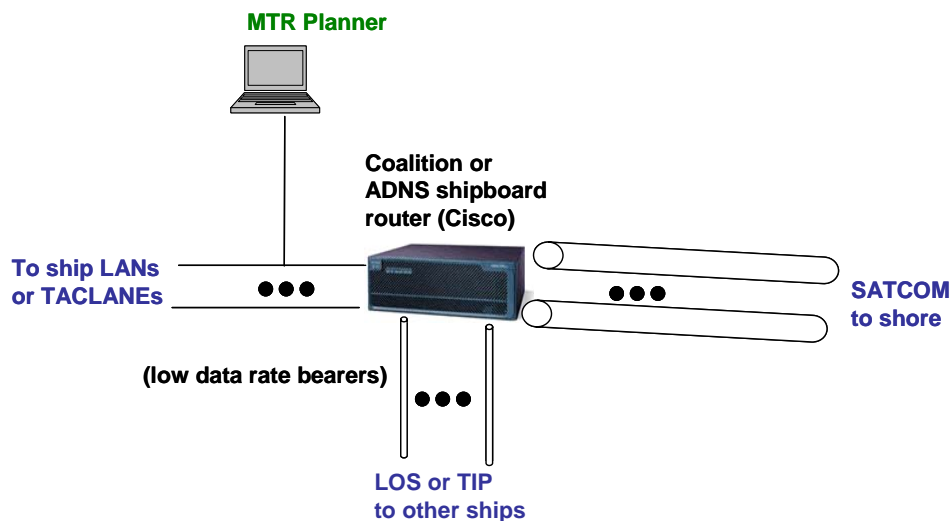


Figure 6-1. Distribution of MTR functional elements in the IANM implementation framework.

Figure 6-1 shows a notional “MTR Planner” that is connected to a Cisco router on a link residing in the same security enclave as the router. The router, in general, is assumed to run OSPFv2, and has three groupings of interfaces:

- 1) SATCOM links to shore; these may be managed by Cisco Optimized Edge Routing (OER) or use static policy-based route maps, and may be in a separate area than the other interfaces;
- 2) line of sight (LOS) or other (EHF TIP) links to other ships; and
- 3) Internal LAN segments, connected to SECRET router or TACLANEs (if an ADNS router).

If the router is not the ADNS router but a separate coalition router, the same conceptual diagram applies, except that the links labelled “SATCOM to shore” are more likely a single link to a TACLANE for tunneling over the ADNS network.

The operational problem is to make effective use of the LOS links by spreading application traffic appropriately. The technical problem is that OSPFv2 does not allow load-balancing across the links unless they are configured to be of equal cost, and equal-cost multipath does not work across multiple areas (both ADNS and Coalition network configurations are planned to place LOS links in a different OSPF area than the Satcom link). Therefore, solutions within the existing OSPFv2 framework alone are limited.

Further, it is important to note that the Satcom links have more bandwidth typically than the LOS links (which have burst rates less than 10Kb/s on average per ship), but the use of Satcom resources to carry routine ship-to-ship traffic is expensive. In the figure, this bandwidth mismatch is depicted by the diameter of the conceptual pipes.

Another requirement is for dynamic response to topological changes and to transient congestion periods. Topological changes can be detected by OSPF mechanisms. Transient congestion is more difficult to passively detect, because these links are connected via FastEthernet to other communications equipment. For instance, the HF-IP link is actually a separate Linux router running OSPF with an HF-IP interface. Moreover, there may be another router between the HF-IP and the Cisco router, such as a Checkpoint firewall running OSPF. This is significant because the Cisco router depicted above may not have a good view of congestion over these links by observing congestion in its output queues, because the congestion may occur in a downstream queue.

We assume that each ship is responsible for managing its own communications devices, and that it does not have authority to change configuration on either the shore’s or other ship’s routers.

There is a requirement to allow the operator to define static policy-based route maps or other routing policy, to control the routing beyond the paths found by the default dynamic routing protocol. These include the definition of

preferences for certain applications to traverse or avoid certain link types, for policy or QoS reasons. Furthermore, operators may want to deploy firewalls and other WAN optimizers in the network, at multiple points along the end-to-end path, and may want to define queueing configurations that help to implement the drop policies during times of overload.

To summarize the goals

- OSPFv2 on Cisco 2800- and 3800-series routers is assumed as the baseline routing environment.
  - Links may be configured in multiple OSPF areas.
  - The network may additionally have layer-3 firewalls (with routing capability) and may have additional WAN traffic optimizers situated between the Cisco routers and the LAN servers/hosts.
- Load-balance application data as appropriate, over all available links. Specifically:
  - application data should be sent over a path that suits its QoS (latency) requirements as best as possible;
  - traffic must be identified via transport port numbers and/or DSCP code point values;
  - when the offered load exceeds the total path capacity between destinations, traffic should be rerouted in a prioritized fashion;
  - operators should be able to define policy rules that prevent traffic from traversing certain types of links;
  - operators should be able to define policy rules that cause traffic to flow preferentially over certain types of links.
- Only local router configuration may be affected; therefore, it is recognized that inbound load balancing is the responsibility of other ships, and that such a solution is not globally optimal and may lead to asymmetric routing.
- Congestion must be detected and responded to in an automatic fashion even though it may occur in a downstream queue that is not directly accessible by monitoring software.
- Topology failures must be responded to in an automatic fashion; OSPFv2 mechanisms for detecting topology failures (loss of HELLO packets) are considered adequate in terms of response time.

The MTR Planner as part of CLIN 0004 was designed to meet these goals and did in fact meet them all to some degree.

## 6.2 Software Overview

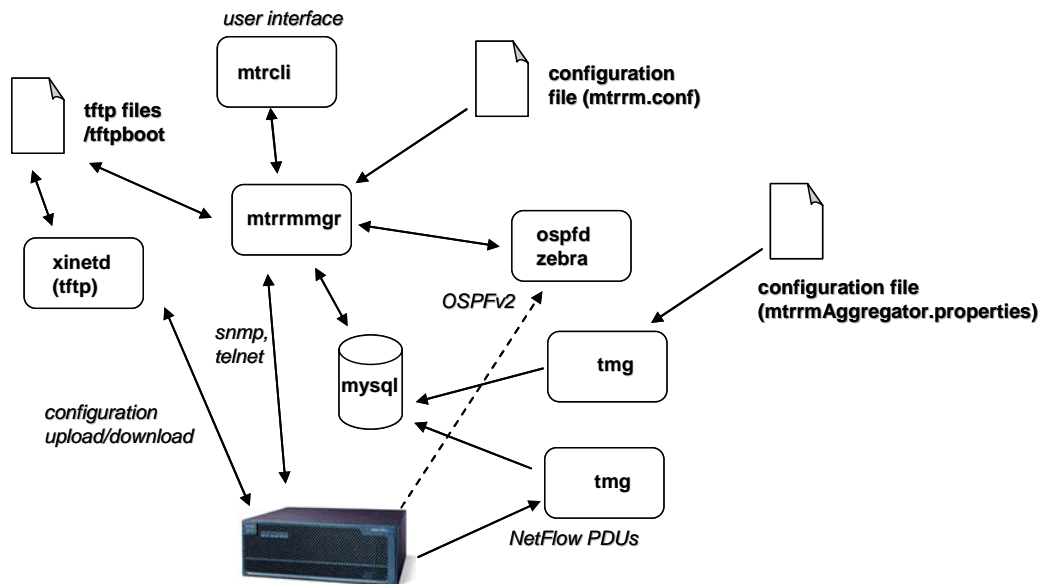


Figure 6-2. Internal software organization

Figure 6-2 illustrates the main functional division of the software. When running, there are five main software processes on the MTR Planner, as well as an instance of a MySQL database:

- **MTR Route Manager:** The MTR Route Manager (mtrmmgr) is written in C++ and interacts with the OSPFv2 daemon, the MySQL database, the MTR Client Interface (mtrcli), and the Cisco router. This process implements the main state machine for the MTR Planner that monitors the Cisco router and overall routing topology and dynamically inserts and removes route maps. The main configuration file for the system is the mtrrm.conf file.
- **MTR Client Interface:** The MTR Client Interface (mtrcli) is written in C++ and interacts with the MTR Route Manager to provide a menu-based interface to the user.
- **NetFlow Collector:** This Java process captures NetFlow PDUs from the Cisco router and writes them to the MySQL database.
- **NetFlow Aggregator:** This Java process connects to the MySQL database to periodically fetch raw NetFlow flow records and aggregate them, and writes the resulting aggregated records (traffic matrix) back to the database. The periodicity of this process is controlled by the file mtrrmAggregator.properties.
- **OSPFv2 daemon in passive mode:** The OSPFv2 daemon (ospfd) is from the quagga routing suite, and is written in C. It is specially compiled to operate in passive mode, which means that it only collects the LSA database across its adjacency with the router but does not advertise routes or install any OSPF routes in its own forwarding table.



### 6.3 Trident Warrior 2008 LOE Testing

#### TW08 RRLOE Configuration SPAWARSSCEN San Diego 14-25 January 2008

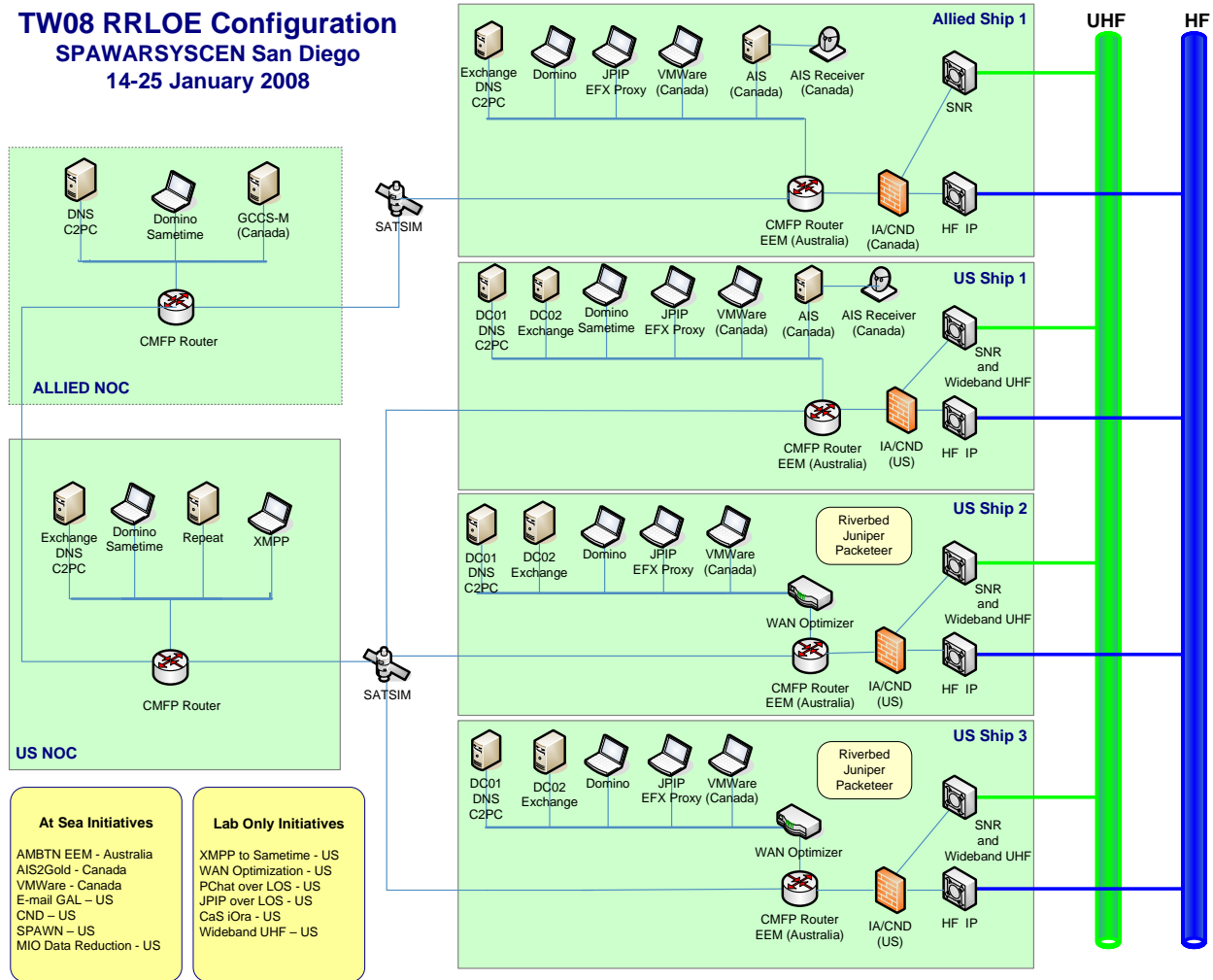


Figure 6-3. Planned TW08 LOE configuration.

Figure 6-3 depicts a recently conducted Trident Warrior `08 (TW08) Limited Objective Experiment (LOE) configuration including the use of four ships, two NOCs, two satellite simulators, and two low-data-rate LOS bearers. At the time of this writing, some items are uncertain regarding the eventual Trident Warrior `08 configuration:

- It is not clear where the firewalls depicted will be deployed (on LOS side of routers, on LAN side, neither, or both);
- It is not clear for which configurations the WAN optimizers will be tested;
- Not all hosts/applications depicted will be tested; and
- Not all “Initiatives” identified will be tested

This diagram therefore represents somewhat of a maximal set of what might be covered in Trident Warrior `08, as projected in late fall 2007.

At the invitation of Dr. Stephan Lopic (SPAWAR SSC-SD), we participated in some initial testing of the MTR-RM software from January 14-16, 2008, in San Diego. We brought a Linux laptop running MTR-RM software, and connected it to one of the Cisco routers. The routing configuration used multiple OSPF areas and redistribution of connected links into OSPF, and routed all ship-to-ship traffic by default over the satellite link. We demonstrated the

following capabilities:

- Dynamic reroute of (unmarked) application traffic, identified by transport port number. The two applications tested were C2PC and email.
- Dynamic monitoring of congestion on the outbound Cisco interfaces, and reporting of the dynamic traffic state via Cisco NetFlow and SNMP
- Automated removal of the policy-based route map when the HF-IP link emulator (audio mixer) was disabled

## 7. Results, Discussion, and Conclusions

This document has summarized the Multi-Topology Routing (MTR) Route Manager and Analyzer (RMA) program for The Boeing Company under Office of Naval Research (ONR) Contract N00014-05-C-0012. The MTR RMA program ran from July 2005 through February 2008, and was part of a broader Integrated Autonomous Network Management (IANM) program funded by ONR and coordinated by Navy SPAWAR Systems Center, San Diego.

Boeing's program made the following contributions:

1) developed research software, known as the MTR Planner, that can perform basic load balancing of an OSPF-based network, given a set of traffic inputs on the system. We developed techniques to leverage Cisco NetFlow information to dynamically determine the traffic matrix imposed on a network. We also developed interfaces and techniques to allow the MTR Planner to passively observe routing updates in a network, and to directly interact with Cisco routers to upload configuration changes and download network management information. Because Cisco routers unexpectedly did not support OSPF MTR routing during the timeframe of this contract, we developed a workaround that allowed our system to produce route maps that functionally replaced the MTR routing capability. Finally, we performed the system engineering necessary to integrate all of the software into running demonstration systems.

We developed two variants of this software: a *centralized planner* integrated with other IANM software, and a *distributed planner* decoupled from other IANM software.

- The *centralized planner* provided an interface to allow mission plans to be distilled into low-level routing configurations deployed globally in the network (to IANM agents operating on individual ships). We developed a capability that not only supported this off-line planning scenario but also could be used in a real-time network monitoring and maintenance capacity. This capability was integrated with the other IANM software components and demonstrated in conjunction with the broader IANM system.
- The *distributed planner* manages a set of outbound links from the router to which it is connected. This version of the software has no other IANM dependencies. It is again designed to work with the Cisco routers being deployed in coalition and ADNS networks. It operates autonomously from ship to ship, which limits the scope of its effectiveness (a given ship can only control its outbound links, and global coordination is not possible) but offers deployment advantages.

2) supported the broader IANM program by participating in two integrated demonstrations at SPAWAR Systems Center, San Diego (March 2006 and August 2006) and in an MTR-specific demonstration in February 2007. In addition, we supported a Trident Warrior 2008 Limited Objective Experiment (LOE) in January 2008 with our distributed planner.

3) supported the MTR experiment as part of the Allied Multi Bearer Routing Networking (AMBTN) initiative during Trident Warrior 2006. Coalition ships were equipped with a Linux-based router running Boeing's MTR software. Results from testing showed that all packets were marked and routed correctly, while the network administrative traffic load was reduced [Sib06].

4) participated in the standardization effort for MTR by contributing to the following Internet-Drafts at the Internet Engineering Task Force (IETF):

- *Multi-Topology (MT) Routing in OSPF* [RFC4915]
- *OSPF Version 2 MIB for Multi-Topology (MT) Routing* (Internet Draft: draft-ietf-ospf-mt-mib-01.txt) [Raw07]

5) published a technical paper in IEEE Milcom 2007 conference: *Traffic Engineering with OSPF Multi-Topology Routing* [Bae07].

In 2008, Boeing participated in a Trident Warrior 2008 Limited Objective Experiment (LOE) and transferred the software to SSC San Diego, where it may be used in future coalition networking experiments. This transfer was documented in a memorandum dated 10 January 2008 and signed by the IANM, SSC SD, and ONR representatives (see Appendix A).

## References

- [Bae07] K. Bae and T. Henderson, "Traffic Engineering with OSPF Multi-Topology Routing," Proceedings of IEEE Milcom 2007 conference, October 2007.
- [BHK06] K. Bae, T. Henderson, and D. Kushi, "MTR Data Collection Engine Documentation," CLIN 0002 of ONR Contract No. N00014-05-C-0012, September 2006.
- [BHK06b] K. Bae, T. Henderson, and D. Kushi, "Software Design Document," CDRL A003 of ONR Contract No. N00014-05-C-0012, June 2006.
- [BHK07] K. Bae, T. Henderson, and D. Kushi, "MTR Basic Analysis Documentation," CLIN 0003 of ONR Contract No. N00014-05-C-0012, March 2007.
- [Boe05] Integrated Autonomous Network Management Multi-Topology Route Manager and Analyzer, ONR Contract No. N00014-05-C-0012, 9 June 2005.
- [Boe06] "Software Design Document," CDRL A003, Integrated Autonomous Network Management Multi-Topology Route Manager and Analyzer, ONR Contract No. N00014-05-C-0012, June 2006.
- [Cas04] R. Casey, "ADNS Transition Approach to the Future GIG Black Core (Draft)," SPAWAR ADNS Program Office Document, 1 December 2004.
- [Hen08] T. Henderson et al., "MTR Basic Analysis with Policy Constraints Documentation," CLIN 0004 of ONR Contract No. N00014-05-C-0012, February 2008.
- [HKB+06] T. Henderson, D. Kushi, K. Bae, and J. Kim, "Multi-Topology Routing Concept of Operations," IANM internal technical document, 2006.
- [IAN05] "Integrated Autonomous Network Management (IANM) Concept of Operations (CONOPS)," prepared by SYS Technologies, Version 3.1, 31 October 2005.
- [RFC2328] J. Moy, "OSPF version 2," Internet RFC 2328, April 1998.
- [RFC4915] P. Psenak et al., "Multi-Topology (MT) Routing in OSPF," Internet RFC 4915, June 2007.
- [Raw07] N. Rawat et al., "OSPF Version 2 MIB for Multi-Topology (MT) Routing," Internet-Draft: draft-ietf-ospf-mt-mib-01, August 2007.
- [Sib06] R. Sibbald, "TW06 AZ Hotwash," email to experimentation@spawar.navy.mil distribution list, July 5, 2006.

This page intentionally left blank.

10 January 2008

**Memorandum for the Record**

**SUBJ: Multi-Topology Routing (MTR) Application to Coalition Networks**


In 2005, Boeing began work on the application of Open Shortest Path First (OSPF) Multi-Topology Routing (MTR) to Navy scenarios. At this time, MTR was a component of the Office of Naval Research (ONR) FORCENet Future Naval Capability (FNC) Project Integrated Autonomous Network Management (IANM). MTR is envisioned to provide Navy network operators with better traffic management capabilities.

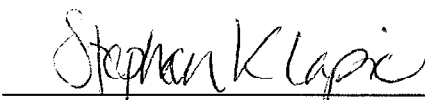
In 2007, Boeing, Space and Naval Warfare Systems Center San Diego (SSC San Diego), and ONR agreed to decouple MTR from IANM based on:

- Lack of Cisco MTR functionality
- Emerging Automated Digital Network System (ADNS) Increment III design
- Discussions and feedback from ADNS engineers

Boeing tailored its remaining work on MTR extensions to OSPF to the management of Line of Sight (LOS) and extended Line of Sight (ELOS) links. These extensions were previously demonstrated as a coalition initiative during Trident Warrior 06 on the CENTRIXS network using Linux routers. With these extensions still unavailable in the Cisco Internet Operating System (IOS) for 3800-series routers and below, Boeing subsequently sought to approximate its MTR functionality using dynamic route maps to be employed with Cisco routers. The route map (RM) surrogate solution is called MTR-RM. The development of this capability is targeted for the management of LOS/ELOS links for the Trident Warrior 08 coalition network.

The Boeing contract concludes in February 2008. The remaining program plan for Boeing is to participate in the Trident Warrior 08 coalition Limited Objective Experiment (LOE) to be conducted January 2008. Upon conclusion of the LOE, the MTR software, both OSPF extensions and MTR-RM, will be transferred to SSC San Diego.

 1/14/08  
Jan Myers, IANM Principle Investigator SSC SD Code 5525

 1/14/08  
Stephan Lopic, Chief Engineer, Networks Division SSC SD Code 55106

  
John Kuchinski, Consultant to ONR Code 311 Program Office